REMARKS

Claims 1 - 30 are pending in the application. Claims 1- 30 have been rejected.

Claims 10, 20 and 30 stand rejected under 35 U.S.C. § 112, first paragraph, because "neither the specification or the claims defined what it meant to be in an 'input/output state', so it is impossible for someone skilled in the art to enable the invention at stated". (Office action, Page 2, ¶ 1.) The language of claims 10, 20 and 30, which is now incorporated into independent claims 1, 11 and 21, respectively, has been amended to recite "a pager input/output state". The term "pager I/O state" is discussed throughout the Hepkin application.

Claims 1 - 30 stand rejected under 35 U.S.C. § 102(b) as being anticipated by Stoodley, U.S. Patent No. 6,408,305 B1 (Stoodley).

The present invention, as set forth by independent claim 1, relates to a method for initializing a memory page, the method includes in response to a memory operation by a first thread, allocating a memory page, generating a request for a second thread to initialize the allocated memory page, initializing the allocated memory page by the second thread in accordance with the request, indicating the memory page as being in a pager input/output state after allocating the memory page, and indicating the allocated memory page as being in a normal state after initializing the memory page.

The present invention, as set forth by independent claim 11, relates to a computer program product on a computer readable medium for use in a data processing system for initializing a memory page, the computer program product includes means for allocating a memory page in response to a memory operation by a first thread, means for generating a request for a second thread to initialize the allocated memory page, means for initializing the allocated memory page by the second thread in accordance with the request, means for indicating the memory page as being in a pager input/output state after allocating the memory page and means for indicating the allocated memory page as being in a normal state after initializing the memory page.

The present invention, as set forth by independent claim 21, relates to an apparatus for initializing a memory page, the apparatus includes means for allocating a memory page in response to a memory operation by a first thread, means for generating a request for a second thread to initialize the allocated memory page, means for initializing the allocated memory page by the second thread in accordance with the request, means for indicating the memory page as being in a pager input/output state after allocating the memory page and means for indicating the allocated memory page as being in a normal state after initializing the memory page.

Stoodley discloses providing access frontier pages around all accessible memory pages loaded from an object-oriented database to prevent simultaneous access by multiple threads to an initializing page. When discussing "frontier" pages, Stoodley sets forth that a collection of inaccessible pages that surround a root page form an access frontier for the root page. (See e.g., Stoodley Co. 6, lines 44 – 51.)

On initializing the root pages from an object-oriented database, access frontier pages corresponding to each swizzled pointer are initialized and marked as inaccessible. All pointers contained in these access frontier pages that do not point to either an initialized and accessible page such as a root page or another access frontier page have page table entries created for them and are marked inaccessible.

The Examiner cites to the following portion of Stoodley when setting forth that Stoodley discloses indicating a memory page as being in an input/output state after allocating the memory page:

> Upon occurrence of the fault, the page fault interrupt handling routine of the present invention proceeds, for each of the uninitialized pages pointed to by pointers on the faulted page [620], to find a free page frame in main memory (taken from a free page frame list maintained by the page protection system 144 or "paging" out an occupied page to storage to make a free page frame available in main memory 140) for the page table entry previously allocated to that uninitialized page (note that all pointers in access frontier pages not pointing to a root page or other access frontier pages were swizzled but no backing memory was allocated to the pages corresponding to those pointers viz. object 4 page 440), transfer the corresponding page from the object server 220 into the page frame allocated in main memory 140 via the persistent object system, modify a page table in the page protection system 144 to reflect that the page is now in memory and swizzle the pointers in the page, all resulting in an access frontier page [630].(Stoodley, Col. 7, lines 36 – 53).

The Examiner cites to the following portion of Stoodley when setting forth that Stoodley discloses indicating the allocated memory page as being in a normal state after initializing the memory page:

> So when the process of initializing the access frontier pages is complete, the access frontier pages are fully initialized, including complete swizzling of all pointers contained in them. A deference operation that makes access to an object contained in one of the access frontier pages, requires simply marking that page as accessible subject to the completion of access frontier page processing of the present invention (Stoodley Col. 6, lines 58 – 65).

However, nowhere in the cited portions of Stoodley, or anywhere else in Stoodley, is there a disclosure or suggestion of indicating a memory page as being in a pager input/output state after allocating the memory page or indicating the allocated memory page as being in a normal state after initializing the memory page as claimed in amended independent claims 1, 11 and 21.

More specifically, Stoodley, taken alone or in combination, does not teach or suggest a method for initializing a memory page, where the method includes indicating the memory page as being in a pager input/output state after allocating the memory page, and indicating the allocated memory page as being in a normal state after initializing the memory page, all as required by claim 1. Accordingly, claim 1 is allowable over Stoodley. Claims 2 - 10 depend from claim 1 and are allowable for at least this reason.

Stoodley, taken alone or in combination, does not teach or suggest a computer program product on a computer readable medium for use in a data processing system for initializing a memory page where the computer program product includes means for indicating the memory page as being in a pager input/output state after allocating the memory page and means for indicating the allocated memory page as being in a normal state after initializing the memory page, all as required by claim 11. Accordingly, claim 11 is allowable over Stoodley. Claims 12 - 20 depend from claim 11 and are allowable for at least this reason.

Stoodley, taken alone or in combination, does not teach or suggest an apparatus for initializing a memory page where the apparatus includes means for initializing the allocated memory page by the second thread in accordance with the request, means for indicating the

memory page as being in a pager input/output state after allocating the memory page and means for indicating the allocated memory page as being in a normal state after initializing the memory page, all as required by claim 21. Accordingly, claim 21 is allowable over Stoodley. Claims 22 - 30 depend from claim 21 and are allowable for at least this reason.

CONCLUSION

In view of the amendments and remarks set forth herein, the application is believed to be in condition for allowance and a notice to that effect is solicited. Nonetheless, should any issues remain that might be subject to resolution through a telephonic interview, the examiner is requested to telephone the undersigned.

I hereby certify that this correspondence is being electronically submitted to the COMMISSIONER FOR PATENTS via EFS on November 14, 2006.

/Stephen A. Terrile/

Attorney for Applicant(s)

Respectfully submitted,

/Stephen A. Terrile/

Stephen A. Terrile
Attorney for Applicant(s)
Reg. No. 32,946